

Elements of Bioinformatics

Autumn 2011



Juho Rousu

**[http://www.cs.helsinki.fi/courses/
582606/2011/s/k/1](http://www.cs.helsinki.fi/courses/582606/2011/s/k/1)**

Lecture Mon 31.10.



GENE PREDICTION

Molecular biology concepts recap



Nucleotides A, C, G, T

gene

DNA

...TACCTACATCCACTCATC...AGCTACGTTCCCCGACTACGACATGGTGATT

5' ...ATGGATGTAGGTGAGTAG...TCGATGCAAGGGGCTGATGCTGTACCACTAA... 3'

exon

intron

exon

RNA

...AUGGAUGUAGAUGGGGCUGAUGCUGUACCACUAA

codon

transcription

regulation

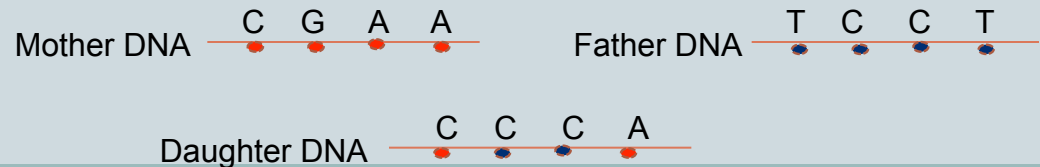
translation

Protein

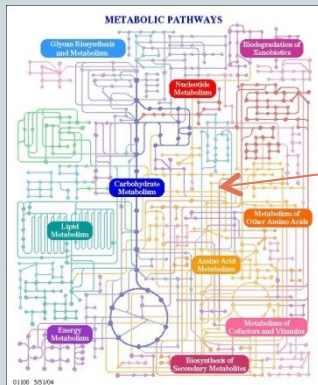
MDVDGLMLYH

regulation

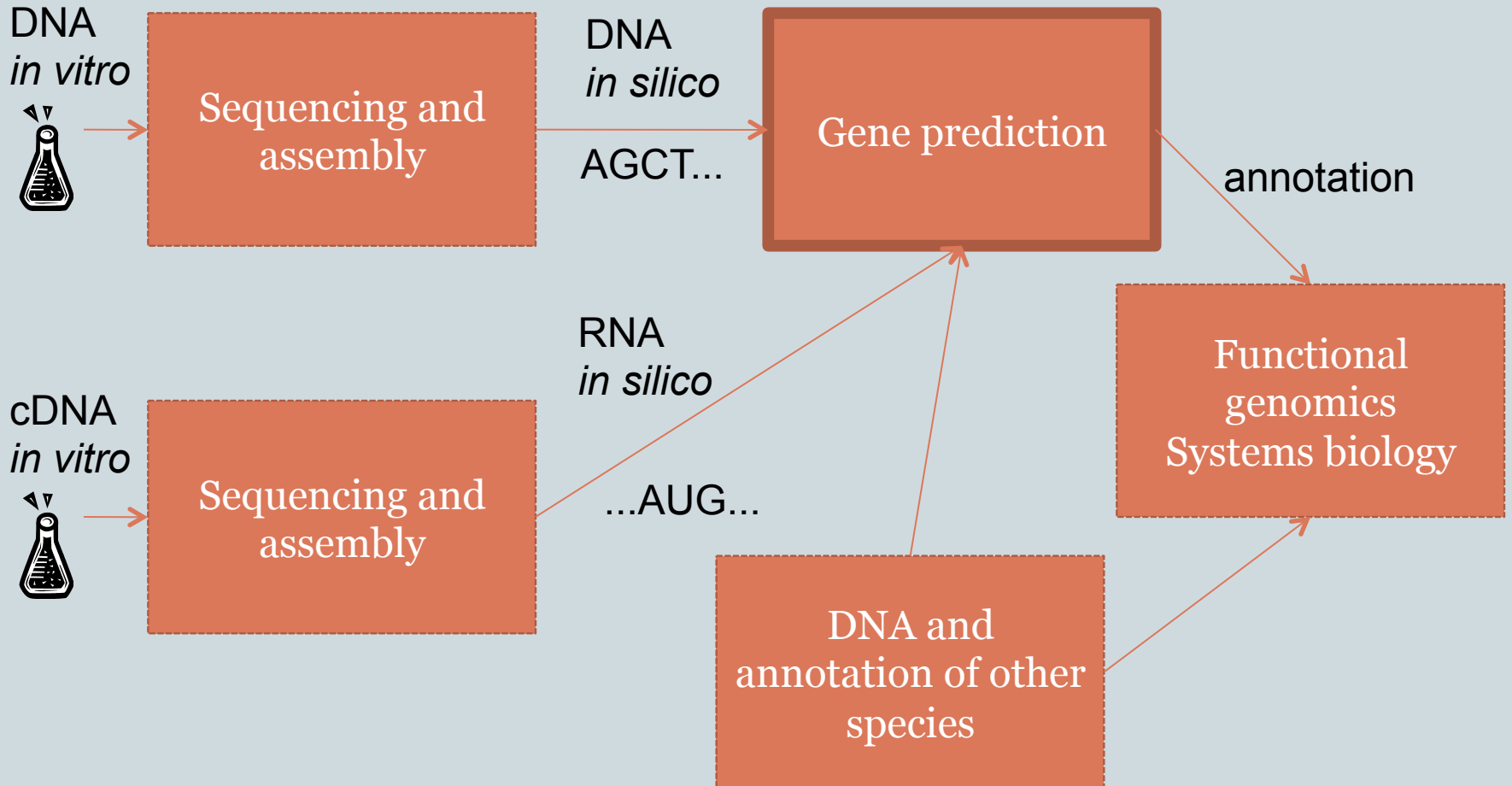
recombination



enzyme



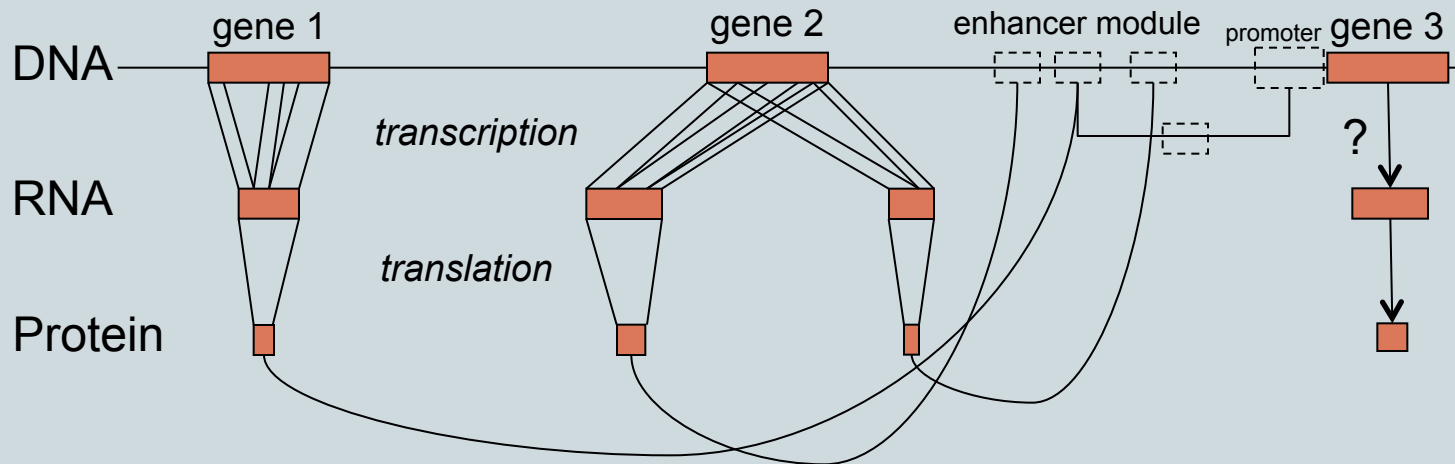
Genome analysis pipeline



Gene structure



- Genes
 - start and stop codons
 - exons, introns (in eukaryotic organisms)
- Promoter regions
 - binding sites for regulatory proteins



Typical eukaryotic gene



- ATG –start codon, TAA –stop codon
- yellow: exons, blue: introns, red: untranslated region
- black: upstream (promoter) and downstream regions

```
12854400 tcaaagtaagttagataaacatgatcattcacaggtcagatggttttaaaaaaaatcattatggtgtacatcacatgtagacaatacttcagaattcattc
12854200 taggaaaagttaatggttacgccaatcacttttttaacagccaacaacatataattagctccaataatcattttttcccctagaatattctcaacct
12854000 attgtccactcaaaacgtgacaaatggaggtctaaagggagaccatacttgactcatttttagagctaggatcagacagagtagatTTTTTgccataactc
12853800 cttgtaaatgtattcacatttcattcccaagaaaaatagactgatgaagaaatataatcagatatgacaaggccgtgctgtttaggttacgtaactctaca
12853600 aggttttaggtctcaatataaacacacaaaagcagatagaagaagcaaacattcacaatcagacaATGACATCTCTCCATACGTTACTCTTCTCTTCTCT
12853400 TCTTTTCTTCATCGTCTTTCCAACCTTCACGTTTCCCTCCACCTTATTGTTTCAGgttcgctcttttagttttgcttcttttacatacacagactctacacac
12853200 tcacttattgggttttctttcaattgtgaaacagAGTTTCAATTGGGAGTCATGGAAGAAAGAAGGAGGATTCTACAATTCTCTCCACAACCTCCATTGACG
12853000 ACATAGCCAACGCTGGAATCACTCATCTTTGGCTTCCCTCCTCTTCTCAATCCGTTGCTCCTGAAGgttccatttctgctttactctttacacattcaca
12852800 tacciaatcttgttactcagcaatcttcaatcctcagGTTACTTACCGGAAAGCTATACGATCTAAACAGCTCCAAATACGGTTCAGAGGCGGAACTGA
12852600 AATCGTTAATCAAAGCGTTGAATCAAAAAGGAATAAAAAGCTTTGGCTGATATAGTGATTAACCCACAGAACAGCTGAGAGGAAAGACGATAAAATGTGGATA
12852400 CTGTTAATTTGGAAGGTGGGACTTCCGATGATCGTCTTGATTGGGATCCTTCTTGTCTGCCGCAATGACCCTAAATTTCCCGGTACCGGAAACCTCGAC
12852200 ACCGGAGGAGATTTTGATGGAGCGCCCGACATCGACCACCTTAACCCTAGAGTTCAGAAAAGAGTTGTCCGAATGGATGAATTGGCTTAAAACCTGAAATCG
12852000 GATTCCATGGTTGGAGATTTGATTATGTTTCGAGGTTATGTCATCTTCCATACCAAAATTTATACGTTTCAGgtaaatcacatatgaattctcaaatatcagac
12851800 aacagtattagtatataagaaacataggttgagataattattactattagtatataagtatcataggttgatagggttatttactactatttagtat
12851600 ataagaacataagtcaatgcaatcaataagaaatataagaagttcactactgattatgtgataaattcctctgtttttggatacacagAATACATC
12851400 ACCGGATTTTGGCGTGGGTGAGAAATGGGACGATATGAAGTACGGAGGAGACGGGAAACTAGACTATGATCAGAACGAGCATCGGTTCGGGTCTCAAACAG
12851200 TGGATCGAGGAAGCGGGTGGTGGTGTGTTGACAGCTTTTGTATTCCACCACCAAGGGGATCTTACAGTCTGTCTCAAAGGTGAGCTTTGGAGACTAAAGG
12851000 ACTCGCAGGGAAAACCGCCTGGTATGATAGGAATCATGCCCGAAACGCTGTACATTCATAGATAACCATGATACATTCAGAACGTTGGGTTTTCCCTTC
12850800 TGATAAAGTCTTGCTTGGATACGTTTATATACTTACTCATCCAGGAACCTTGCATTgtaagtatcatttttagtatgtagctatactattttacaactac
12850600 aatcttgttgatagtattttttgttcagTTTTATAAATCATATAGAAATGGGGACTAAAAGAGAGCATCTCAAAGCTGGTGGCTATCAGGAACAAA
12850400 ATGGGATTTGGTAGCACAAGCTGTAAACGATAAAAAGCGGACAGAGCGGATCTTACTTGGCTATGATTTGATGATAAAGTTATCATGAAGATTGGACAAA
12850200 CCAAGATGTGGGAACACTTGTCTTCTAATTTTGCCTTATTAGCTTATTAGCCTTGTACTTTGCTGTCTGGGAGAAGAAGTAAcgcataactcgaatcata
12850000 agaaaagtaatcgaatgtatcttcttcttttaataaaaacattttggcagtatctaaagatatgtataatgaaatataaaaatgataaagaatacctaaa
12849800 taaaaagagcactagtgggtgtaaaggatacaactccagtgaaagaaaagagttcaagtgaaagagtgcaacttgtagaaataagattggaaagtttc
12849600 catcgttttgtttggcatacaactaatatattatatttgccgactcgtataagatttgagccctactaaaatcagaattatgatgtcttaacca
12849400 cacaactgccaanaatcagaacgaattatattatgtagaagaagaaaaaaaagtatggtgggaagtgggaacagttagacaggttaattcgaataaa
```

A
T
4
G
2
5
0
0
0
1

<http://en.wikipedia.org/wiki/File:AMY1gene.png>

Gene prediction: main approaches



- Evidence-based gene finding: identify mRNA sequences expressed by the organism and map them back to the genome
- Ab initio gene prediction: detecting the 'signal' of functional elements via statistical approaches or matching against a database of known motifs
- Comparative genomics approaches: detect conserved DNA regions by comparing a large set of related genomes

Evidence-based gene finding

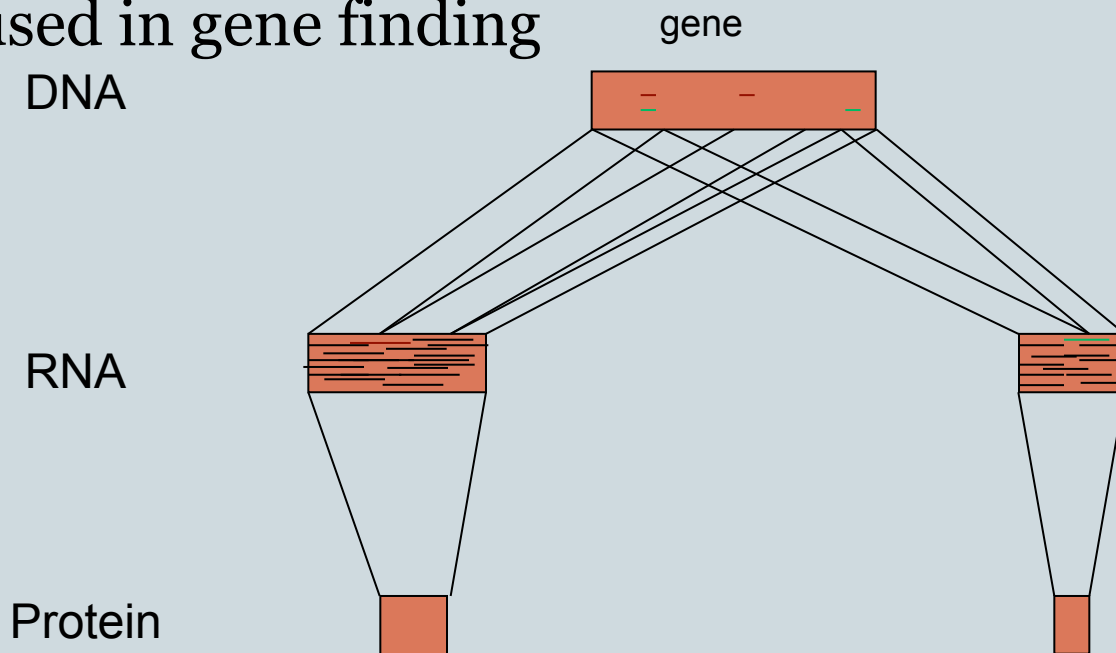


- In evidence-based gene finding, one assumes that there is access to mRNA or protein sequences expressed by the organism
 - RNA-seq is one suitable experimental technique for mRNA
 - Peptide sequencing via tandem mass spectrometry gives amino acid sequences
- Target genome is searched for sequences that match the expressed mRNA or protein sequences
 - Sequence alignment problem using, e.g. BLAST
 - For prokaryotic genes, relatively straight-forwards
 - Exon-intron structure of eukaryotic genes is a complication

RNA-seq for Gene finding



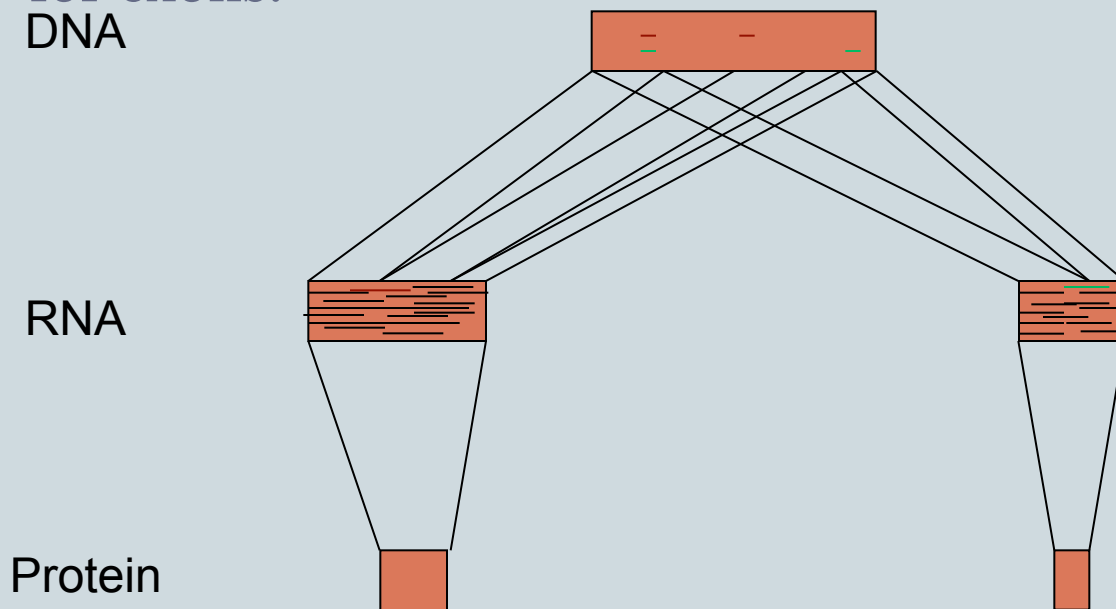
- RNA-seq is a short-read sequencing technology to measure the transcriptome, i.e. all expressed mRNA in a given sample
- Mainly used to study the function of genes, but can also be used in gene finding



RNA-seq for gene finding

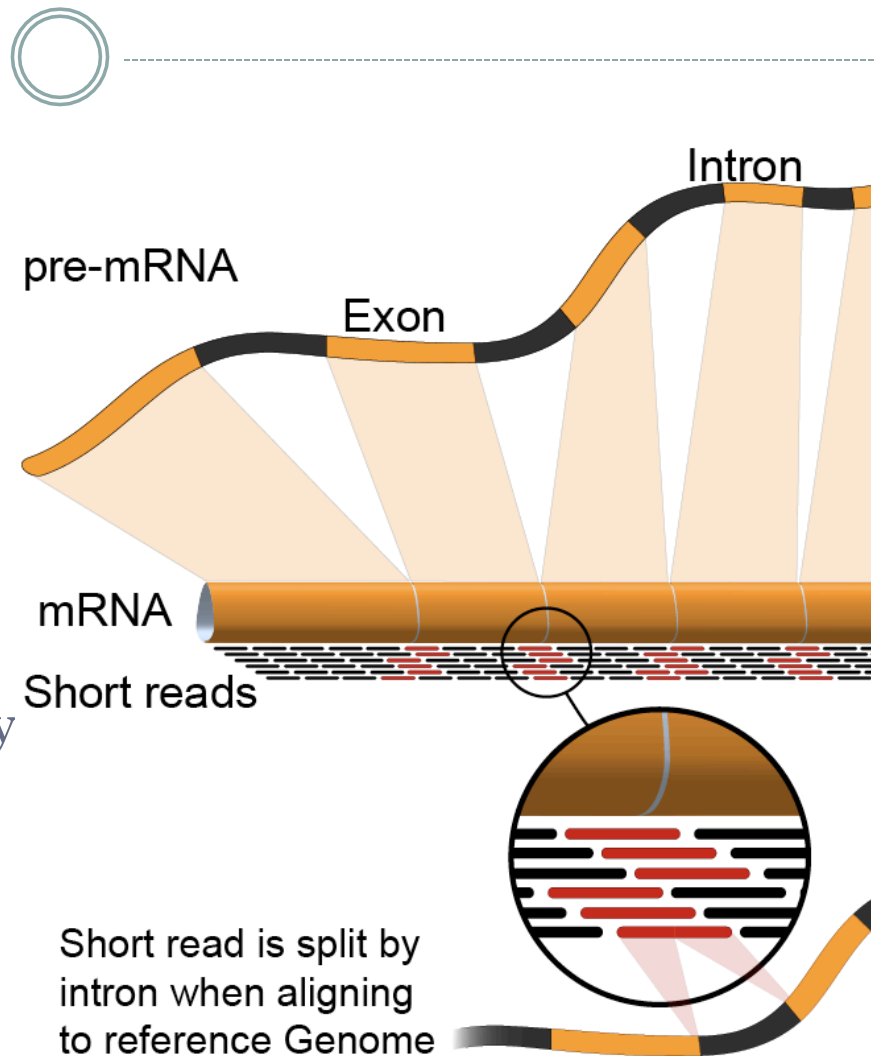


- Two alternative approaches:
 - Assemble mRNA from short reads and match the mRNA transcript to the genome (taking introns into account)
 - Align the short reads of cDNA directly to the genome and vote for exons.



Eukaryotic Gene finding with RNA-seq data

- Consider the first approach: matching the RNA-seq reads to the target genome
- Some reads may cross exon-exon boundary
 - In the target genome these read sequence will be split by an intron
 - Generally we may not know where the exon-exon boundaries are
- Exercises: discover solutions to this problem



Eukaryotic Gene finding with known protein sequences



- Consider matching known protein sequence to the target genome
- As only exons are translated, when matching the protein sequence into the target genome, one needs to consider where the introns might be located
- With dynamic programming one compute a score matrix S , so that $S[i,j,k]$ gives the maximum score of exons in $\text{dna}[1..j]$ translating into $\text{protein}[1..i]$ with k introns
- Exercises: details of the method

Gene finding example

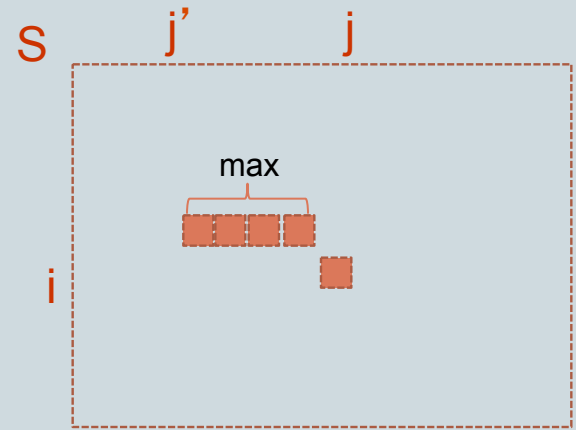


...ATGACATATGTAAATGGTAGCTACGTACGATCGAGGTAGCAGGCCTATTA...^j

MTYVNGSYAYⁱ

match score = 1
mismatch score = 0
indel score = $-\infty$

$S[i,j,0] = -\infty$
 $S[i,j,1] = 10$ (exercise develops the recursion)



...ATGACATATGTAAATGGTAGCTACgtacgatcgaggtagcagGCCTATTA...
 { }

M T Y V N G S Y

{ } { }
A Y

sliding window maxima technique:

3,5,6,2,3,4,7 → 3,5,6,2,3,4,7
 max = 6 max = 7

Enough to store
right-most maxima: $(i, A[i])$ such
 that there is no $A[j] > A[i]$, for indices
 $i < j$ inside the sliding window.

Evidence-based gene finding



- Major limitations of evidence-based approach is coverage
 - mRNA approach:
 - ✦ Not all genes are expressed all the time or all tissues, so mRNA will not in general cover the all genes
 - Known protein approach:
 - ✦ Not all proteins have been sequenced, corresponding genes would be missed
 - ✦ What if the target genome contains previously unknown genes?
- For larger coverage, need ab initio tools that do not require observing the gene products

Ab initio gene prediction



- What can be deduced from just the genome content?
- Introns often have markers at both ends (gt...ag,...), but these markers also appear in other places.
- Statistical properties need to be used to distinguish between coding and non-coding regions.
- Already non-trivial for prokaryotes as not all start codon – stop codon pairs (*open reading frames*, **ORFs**) correspond to genes.
- Hidden Markov Model (HMM) –techniques can be used for this prediction task. (next lecture)

Biological words: k-mer statistics



- To understand statistical approaches to gene prediction, we need to study what is known about the structure and statistics of DNA.
 - 1-mers: individual nucleotides (bases)
 - 2-mers: dinucleotides (AA, AC, AG, AT, CA, ...)
 - 3-mers: codons (AAA, AAC, ...)
 - 4-mers and beyond

1-mers: base composition



- Typically DNA exists as *duplex* molecule (two complementary strands)

5' -GGATCGAAGCTAAGGGCT-3'
3' -CCTAGCTTCGATTCCCGA-5'

Top strand: 7 G, 3 C, 5 A, 3 T
Bottom strand: 3 G, 7 C, 3 A, 5 T
Duplex molecule: 10 G, 10 C, 8 A, 8 T
Base frequencies: 10/36 10/36 8/36 8/36

$$\text{fr}(G + C) = 20/36, \text{fr}(A + T) = 1 - \text{fr}(G + C) = 16/36$$

These are something we can determine experimentally.



G+C content



- $\text{fr}(G + C)$, or *G+C content* is a simple statistics for describing genomes
- Notice that one value is enough to characterise $\text{fr}(A)$, $\text{fr}(C)$, $\text{fr}(G)$ and $\text{fr}(T)$ for duplex DNA
- Is G+C content (= base composition) able to tell the difference between genomes of different organisms?
- Is G+C content able to tell the difference between coding and non-coding regions?

G+C content and genome sizes (in megabasepairs, Mb) for various organisms



• Mycoplasma genitalium	31.6%	0.585
• Escherichia coli K-12	50.7%	4.693
• Pseudomonas aeruginosa PAO1	66.4%	6.264
• Pyrococcus abyssi	44.6%	1.765
• Thermoplasma volcanium	39.9%	1.585
• Caenorhabditis elegans	36%	97
• Arabidopsis thaliana	35%	125
• Homo sapiens	41%	3080

Base frequencies in duplex molecules



- Consider a DNA sequence generated randomly, with probability of each letter being independent of position in sequence
- You could expect to find a uniform distribution of bases in genomes...

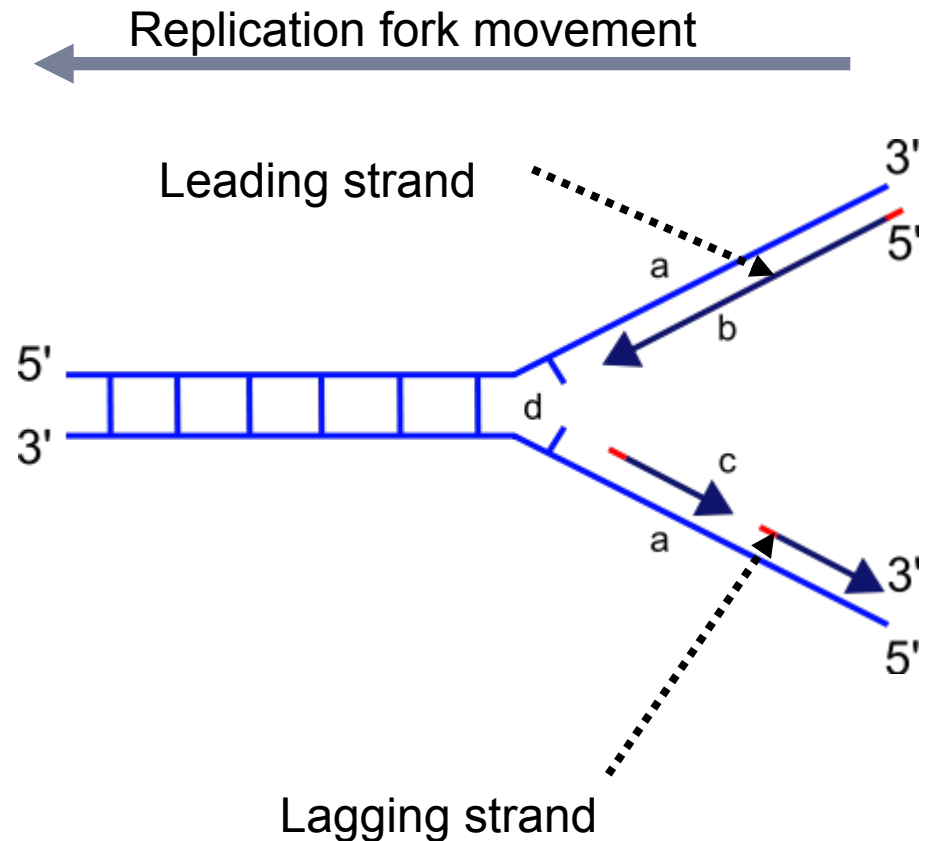
5' - . . . GGATCGAAGCTAAGGGCT . . . - 3'
3' - . . . CCTAGCTTCGATTCCCGA . . . - 5'

- This is not, however, the case in genomes, especially in prokaryotes
 - This phenomena is called *GC skew*

DNA replication fork



- When DNA is replicated, the molecule takes the *replication fork* form
- New complementary DNA is synthesised at both strands of the "fork"
- New strand in 5' -3' direction corresponding to replication fork movement is called *leading strand* and the other *lagging strand*

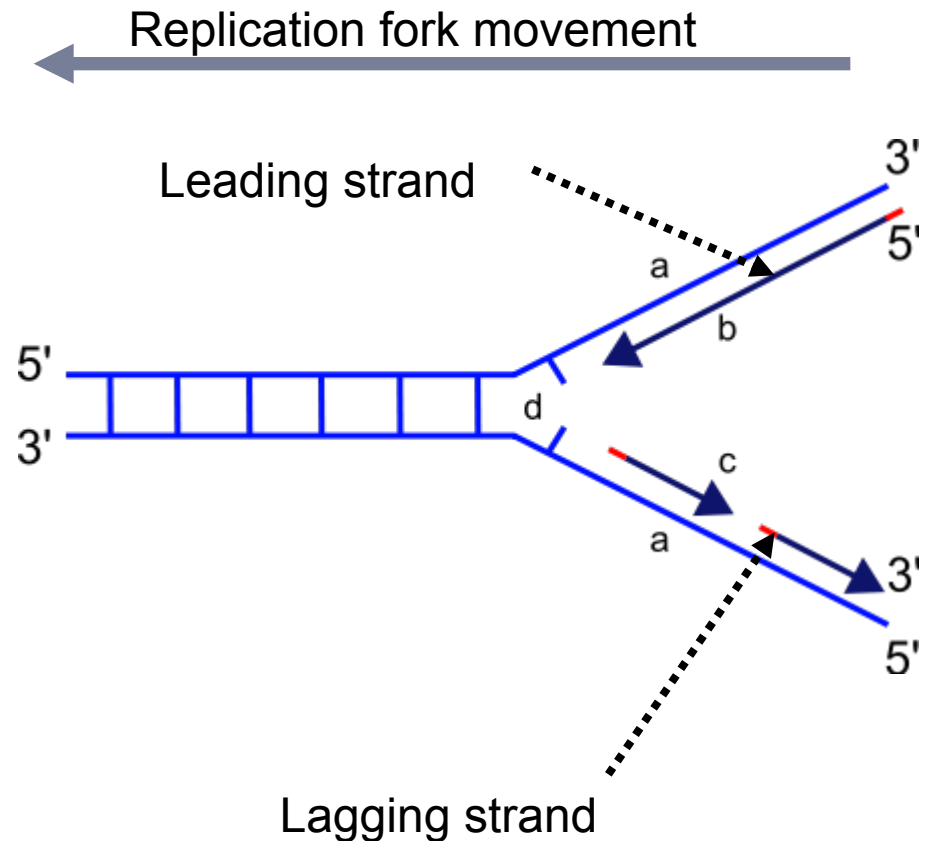


Replication fork

DNA replication fork



- This process has specific starting points in genome (*origins of replication*)
- Observation: Leading strands have an excess of G over C
- This can be described by *GC skew* statistics



Replication fork

GC skew



- GC skew is defined as $(\#G - \#C) / (\#G + \#C)$
- It is calculated at successive positions in intervals (windows) of specific width



$(4 - 2) / (4 + 2) = 1/3$

$(3 - 2) / (3 + 2) = 1/5$

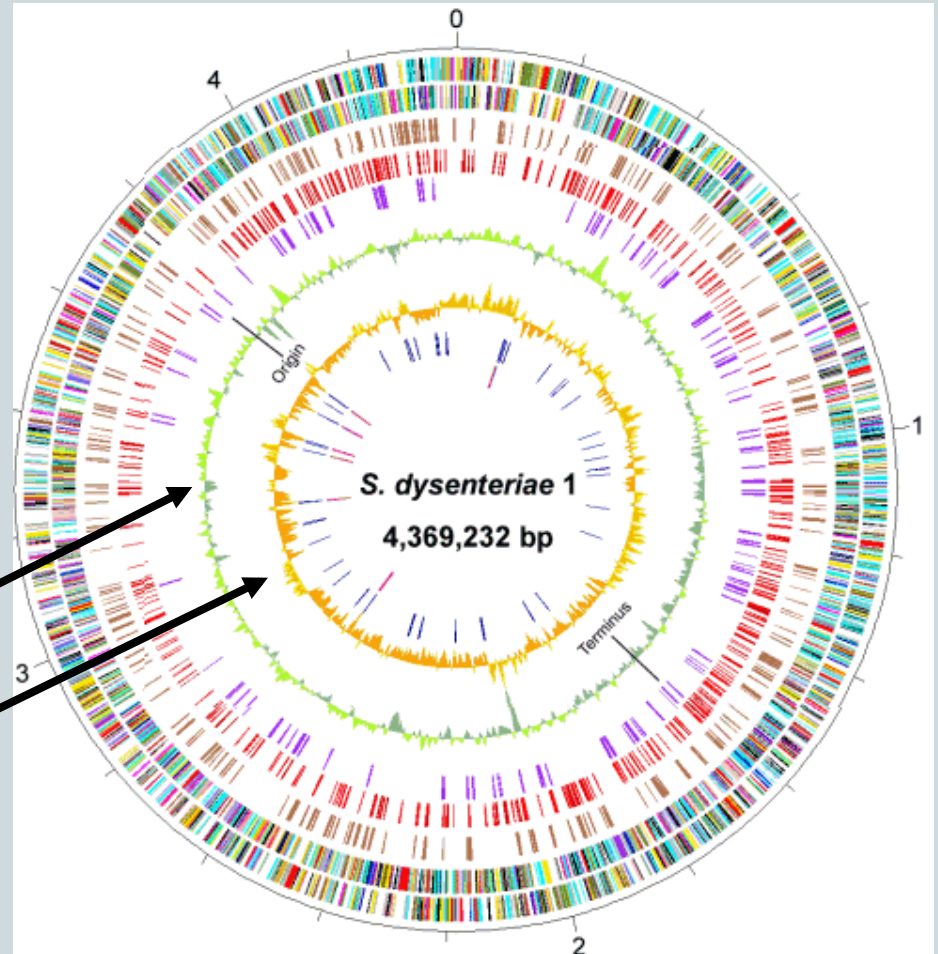
G-C content & GC skew

- G-C content & GC skew statistics can be displayed with a *circular genome map*

G+C content

GC skew

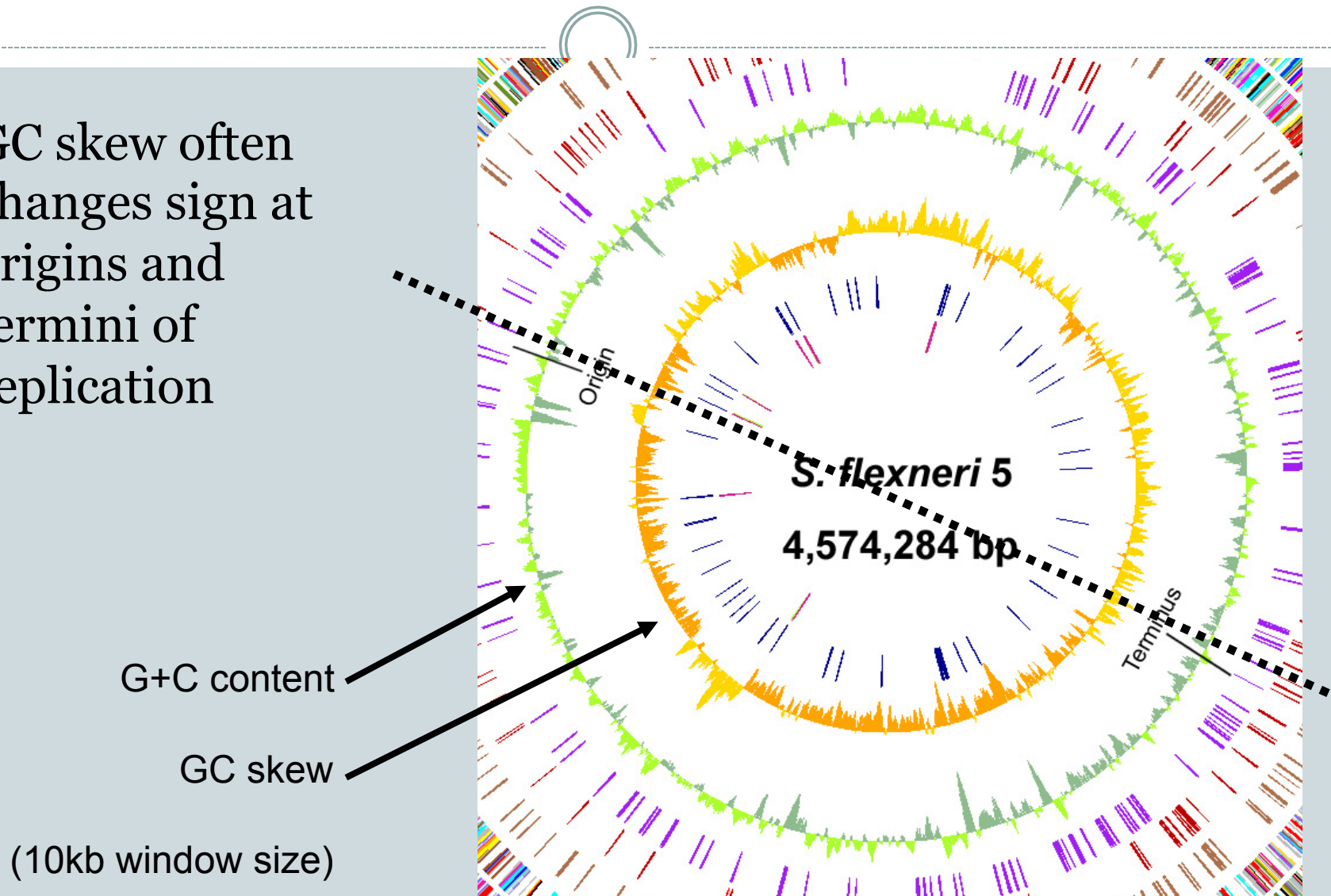
(10kb window size)



Chromosome map of *S. dysenteriae*, the nine rings describe different properties of the genome http://www.mgc.ac.cn/ShiBASE/circular_Sd197.htm

GC skew

- GC skew often changes sign at origins and termini of replication



i.i.d. model for nucleotides



- Assume that bases
 - occur independently of each other
 - bases at each position are identically distributed
- Probability of the base A, C, G, T occurring is p_A , p_C , p_G , p_T , respectively
 - For example, we could use $p_A=p_C=p_G=p_T=0.25$ or estimate the values from known genome data
- Joint probability is then just the product of independent variables
 - For example, $P(TG) = p_T p_G$

Refining the i.i.d. model



- i.i.d. model describes some organisms well but fails to characterise many others
- We can refine the model by having the DNA letter at some position depend on letters at preceding positions

...TCGTGACGCCG?

Sequence context to consider

First-order Markov chains



...TCGTGACGCCG ?

X_t

|

X_{t-1}

- Let's assume that in sequence X the letter at position t , X_t , depends only on the previous letter X_{t-1} (*first-order markov chain*)
- Probability of letter b occurring at position t given $X_{t-1} = a$:
 $p_{ab} = P(X_t = b \mid X_{t-1} = a)$
- We consider *homogeneous* markov chains: probability p_{ab} is independent of position t

Estimating p_{ab}



- We can estimate probabilities p_{ab} ("the probability that b follows a") from observed dinucleotide frequencies

	A	C	G	T
A	p_{AA}	p_{AC}	p_{AG}	p_{AT}
C	$p_{CA} + p_{CC} + p_{CG} + p_{CT}$			
G	p_{GA}	p_{GC}	p_{GG}	p_{GT}
T	p_{TA}	p_{TC}	p_{TG}	p_{TT}

Frequency of dinucleotide AT in sequence

Base frequency $fr(C)$

...the values $p_{AA}, p_{AC}, \dots, p_{TG}, p_{TT}$ sum to 1

Estimating p_{ab}



- $$p_{ab} = P(X_t = b \mid X_{t-1} = a) = \frac{P(X_t = b, X_{t-1} = a)}{P(X_{t-1} = a)}$$

Probability of transition a → b (points to $P(X_t = b \mid X_{t-1} = a)$)

Base frequency of nucleotide a, $fr(a)$ (points to $P(X_{t-1} = a)$)

Dinucleotide frequency (points to $P(X_t = b, X_{t-1} = a)$)

$$0.052 / 0.345 \approx 0.151$$

	A	C	G	T
A	0.146	0.052	0.058	0.089
C	0.063	0.029	0.010	0.056
G	0.050	0.030	0.028	0.051
T	0.086	0.047	0.063	0.140

$$P(X_t = b, X_{t-1} = a)$$

	A	C	G	T
A	0.423	0.151	0.168	0.258
C	0.399	0.184	0.063	0.354
G	0.314	0.189	0.176	0.321
T	0.258	0.138	0.187	0.415

$$P(X_t = b \mid X_{t-1} = a)$$

Simulating a DNA sequence



- From a transition matrix, it is easy to generate a DNA sequence of length n:
 - First, choose the starting base randomly according to the base frequency distribution
 - Then, choose next base according to the distribution $P(x_t | x_{t-1})$ until n bases have been chosen

T T C T T C A A

	A	C	G	T
A	0.423	0.151	0.168	0.258
C	0.399	0.184	0.063	0.354
G	0.314	0.189	0.176	0.321
T	0.258	0.138	0.187	0.415

$$P(X_t = b | X_{t-1} = a)$$

Example Python code for generating DNA sequences with first-order Markov chains.

```
#!/usr/bin/env python
```

```
import sys, random
```

```
n = int(sys.argv[1])
```

} Initialisation: use packages 'sys' and 'random',
read sequence length from input.

```
tm = {'a' : {'a' : 0.423, 'c' : 0.151, 'g' : 0.168, 't' : 0.258},  
      'c' : {'a' : 0.399, 'c' : 0.184, 'g' : 0.063, 't' : 0.354},  
      'g' : {'a' : 0.314, 'c' : 0.189, 'g' : 0.176, 't' : 0.321},  
      't' : {'a' : 0.258, 'c' : 0.138, 'g' : 0.187, 't' : 0.415}}
```

} Transition matrix
tm and initial
distribution pi.

```
pi = {'a' : 0.345, 'c' : 0.158, 'g' : 0.159, 't' : 0.337}
```

```
def choose(dist):
```

```
    r = random.random()
```

```
    sum = 0.0
```

```
    keys = dist.keys()
```

```
    for k in keys:
```

```
        sum += dist[k]
```

```
        if sum > r:
```

```
            return k
```

```
    return keys[-1]
```

} Function choose(), returns a key (here 'a', 'c', 'g' or
't') of the dictionary 'dist' chosen randomly
according to probabilities in dictionary values.

```
c = choose(pi)
```

```
for i in range(n - 1):
```

```
    sys.stdout.write(c)
```

```
    c = choose(tm[c])
```

```
sys.stdout.write(c)
```

```
sys.stdout.write("\n")
```

} Choose the first letter, then choose
next letter according to $P(x_t | x_{t-1})$.

Simulating a DNA sequence



- Now we can quickly generate sequences of arbitrary length...

```
ttcttcaaaataaggatagtgattcttatttggttaagggataacaatntagatctttttcatgaatcatgtatgtcaacggttaaagttgaactgcaataagttc
ttacacacgattggttatctgctgctgcaagcatttcactacatttgccgatgcagccaaaagtatttaacatttggttaacaaattgacttaaatcgcgcaacttaga
gtttgacggttcatagttgatgctgcttaacaattacttttagttttttaaatgctgttctacaatcattaatcagctctggaaaaacattaatgcatttaaac
cacaatggataaattagttacttattttaaaattcacaaagtaattattcgaatagtgccctaagagagtagtgggggttaatggcaagaaaaattactgtagtgaaga
ttaagcctgttattatcacctgggtactctgggtgaatgcacataagcaaatgctacttcagtgtcaaagcaaaaaaattactgataggactaaaaaccctttattt
ttagaatttgtaaaaaatgtgacctcttgcttataacatcatatttattgggtcgttctaggacactgtgattgccttctaactcttatttagcaaaaaattgtcata
gctttgaggtcagacaaaacagtgatggaagacagaaaaagctcagcctagaattagcatgttttgagtggggaattacttgggttaactaaagtgttcatgactgt
tcagcatatgattggttggtgagcactacaagatagaagagttaaactaggtagtgggtgatttcgctaacacagttttcatacaagttctattttctcaatggttt
ggataagaaaaacagcaaaaatttagtatttttctagtaaaaagcaaacatcaaggagaaattggaagctgcttgggtcagtttgcattaaattaaaaaattat
ttgaagtattcgagcaatgttgacagctctgcttctcaaaaagcagcaaatcccctcaaaattgggcaaaaaacctaccctggcttcttttaaaaaaccaagaaa
agtcctatataagcaacaaatttcaaaccttttggttaaaaaattctgctgctgaataaataggcattacagcaatgcaattagggtgcaaaaaaggccatcctcttct
tttttgtaacaattggtcaagcaactttgaatttgagattttaaccactgtctatagggacttcgaattaaattgactgggtctgcatcaciaatttcaactgcc
caatgtaatcatattctagagtattaaaaatacaaaaaagtacaattagttatgccattggcctggcaatttatttactccactttccacgttttggggatatttta
acttgaatagttcacaatcaaaacataggaaggatctactgctaaaagcaaaagcgtattggaatgataaaaaactttgatgtttaaaaaactacaaccttaatgaa
ttaaggtgaaaaaataattcaaaaaagaaattcagttcttgccgagtaaatatttttgatgtttgagatcaggggtacaaaaataagtgcagatgagattaactctcaa
atataaactgatttaagtgtatttgctaataacattttcgaaaaggaatattatggtaagaattcataaaaaatggttaataactgatacaactttcttttatatcctc
catttgccagaatactgttgacacaaactaattggaaaaaaaatagaacgggtcaatctcagtgaggaggagaagaaaaaagtgggtgcaggaaatagtttctacta
acctggtataaaaaacatcaagtaacattcaaatgcaaatgaaaactaacggatctaagcattgattgattttctcatgcctttcgctagttttaataaacgcgc
cccaactctcatcttcggttcaaatgatctattgtatttatgcactaacgtgcttttatggttagcatttttaccctgaagttccgagtcattggcgctcactcacia
atgacattacaatttttctatgttttgctggtgagtcaaaagtcagtcctacaattctttcttatatagaactagacaaaatagaaaaaggcactttggagctc
gaaatgctccttagtttcaaaaaggaaattggtgaatttttggttagttaaattttgaacaaactagtatagtggtgacaaacgatcaccttgagtcggtgacta
taaaagaaaaaggagattaaaaaacctgcggtgccacatttttggttacgggcatttaagggttgcatgtgttgagcaattgaaacctacaactcaataagtcag
ttaagtcacttctttgaaaaaaaagaccctttaagcaagctc
```

Simulating a DNA sequence



Dinucleotide frequencies

Simulated Observed

aa	0.145	0.146
ac	0.050	0.052
ag	0.055	0.058
at	0.092	0.089
ca	0.065	0.063
cc	0.028	0.029
cg	0.011	0.010
ct	0.058	0.056
ga	0.048	0.050
gc	0.032	0.030
gg	0.029	0.028
gt	0.050	0.051
ta	0.084	0.086
tc	0.052	0.047
tg	0.064	0.063
tt	0.138	0.0140

n = 10000

Simulating a DNA sequence



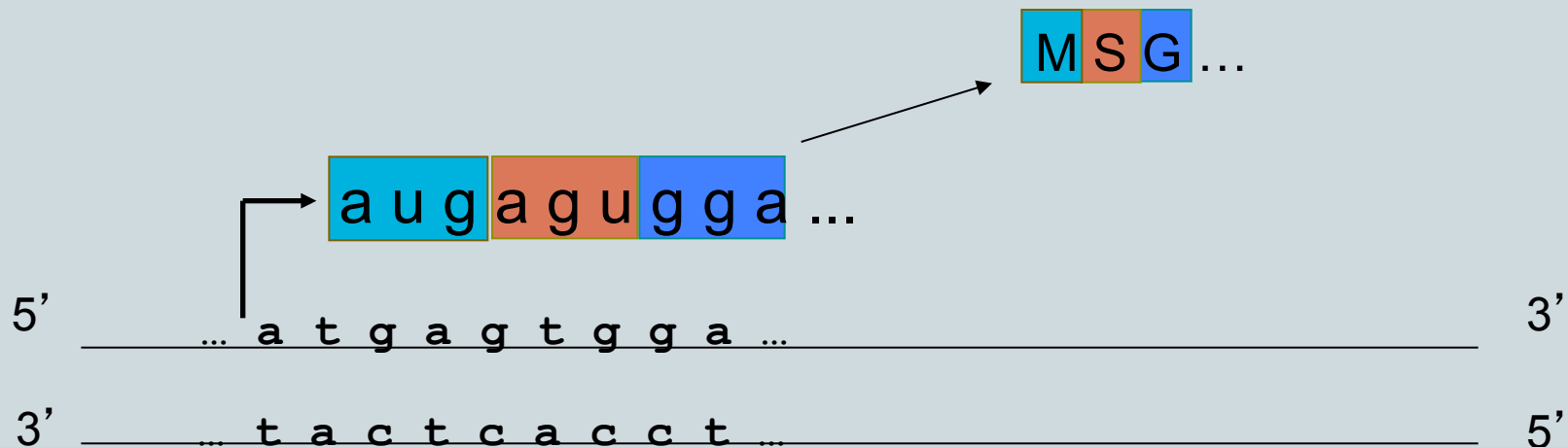
- The model is able to generate correct proportions of 1- and 2-mers in genomes...
- ...but fails with $k=3$ and beyond.

```
ttcttcaaaaataaggatagtgattcttattggcttaagggataacaatttagatcttttttcatgaatcatgtatgtcaacgttaaaaagttgaaactgcaataagttc
ttacacacgattgtttatctgcgtgcgaagcatttcaactacatttgccgatgcagccaaaagtatttaacatttggtaaacaaattgacttaaaatcgcgcaacttaga
gtttgacgtttcatagttgatgctgtctaaacaattacttttagtttttaaatgctgttctacaatcattaatcagctctggaaaaacattaatgcatttaaac
cacaatggataaattagttacttattttaaaattcacaagtaattattcgaatagtgccctaagagagtagctggggttaatggcaaaagaaaattactgtagtgaaga
ttaagcctgttattatcacctgggtactctgggtgaatgcacataagcaaatgctacttcagtgtcaaagcaaaaaaattactgataggactaaaaaccctttattt
ttagaatttgtaaaaaatgtgacctcttgcttataacatcataatttattgggtcgttctaggacactgtgattgccttctaactcttatttagcaaaaaattgtcata
gctttgaggtcagacaaaacaagtgaatggaagacagaaaaagctcagcctagaattagcatgttttgagtggggaaattacttgggttaactaaagtgttcatgactgt
tcagcatatgattgttgggtgagcactacaaaagatagaagagttaaactaggtagtggtgatttcgctaacacagttttcatacaagttctattttctcaatggtttt
ggataagaaaacagcaaaacaaatttagtatttttcttagtaaaaaagcaaacatcaaggagaaaattggaagctgcttgttcagtttgcattaaaataaaaaattat
ttgaagtattcgagcaatgttgacagctctgcgttcttcaaaaagcagcaaatcccctcaaaaattgggcaaaaacctaccctggcttcttttaaaaaaccaagaaa
agtcctatataagcaacaaatttcaaaccttttgttaaaaaattctgctgctgaataaaataggcattacagcaatgcaattagggtgcaaaaaaggccatcctctttct
ttttttgtacaattgttcaagcaactttgaatttgagattttaaccactgtctatatgggacttcgaattaaattgactgggtctgcatcacaaatttcaactgcc
caatgtaatcatattctagagattaaaaatacaaaaaagtacaattagttatgcccattggcctggcaatttatttactccactttccacgttttggggatatttta
acttgaatagttcacaatcaaaaacataggaaggatctactgctaaaagcaaaaagcgtattggaatgataaaaaactttgatgtttaaaaaaactacaaccttaatgaa
ttaaagttgaaaaaataattcaaaaaagaaaattcagttcttggcgagtaataatttttgatgtttgagatcaggggttacaaaaataagtgcagatgagattaactcttcaa
atataaactgatttaagtgtatttgctaataacattttcgaaaaggaatattatggtaagaattcataaaaaatgtttaatactgatacaactttcttttataatcctc
catttggccagaatactgttgacacaaactaattggaaaaaaaatagaacgggtcaatctcagtgaggaggagaagaaaaaagtgggtgcaggaaaatagtttctacta
acctgggtataaaaaacatcaagtaacattcaaattgcaaatgaaaactaacccgatctaagcattgattgatttttctcatgcctttcgcttagtttttaataaacgcgc
cccaactctcatcttcggttcaaatgatctattgtatttatgcactaacgtgcttttatgttagcatttttccacctgaagttccgagtcattggcgtcactcacia
atgacattacaatttttctatgttttgttctgttgagtcaaagtgcagctacaattctttcttatatagaactagacaaaaatagaaaaaggcacttttggagtct
gaaatgccccttagtttcaaaaaaggaaaattgttgaatttttgtgggttagttaaattttgaaacaaactagtatagtggtgacaaaacgatcaccttgagtcggtgacta
taaaagaaaaaggagattaaaaatacctgcggtgccacatttttgttacggggcatttaaggtttgcatgtgttgagcaattgaaacctacaactcaataaagtcatg
ttaagtcacttctttgaaaaaaaagaccctttaagcaagctc
```

3-mers: codons



- We can extend the previous method to 3-mers
- $k=3$ is an important case in study of DNA sequences because of genetic code



3-mers in Escherichia coli genome



Word	Count	Observed	Expected	Word	Count	Observed	Expected
AAA	108924	0.02348	0.01492	CAA	76614	0.01651	0.01541
AAC	82582	0.01780	0.01541	CAC	66751	0.01439	0.01591
AAG	63369	0.01366	0.01537	CAG	104799	0.02259	0.01588
AAT	82995	0.01789	0.01490	CAT	76985	0.01659	0.01539
ACA	58637	0.01264	0.01541	CCA	86436	0.01863	0.01591
ACC	74897	0.01614	0.01591	CCC	47775	0.01030	0.01643
ACG	73263	0.01579	0.01588	CCG	87036	0.01876	0.01640
ACT	49865	0.01075	0.01539	CCT	50426	0.01087	0.01589
AGA	56621	0.01220	0.01537	CGA	70938	0.01529	0.01588
AGC	80860	0.01743	0.01588	CGC	115695	0.02494	0.01640
AGG	50624	0.01091	0.01584	CGG	86877	0.01872	0.01636
AGT	49772	0.01073	0.01536	CGT	73160	0.01577	0.01586
ATA	63697	0.01373	0.01490	CTA	26764	0.00577	0.01539
ATC	86486	0.01864	0.01539	CTC	42733	0.00921	0.01589
ATG	76238	0.01643	0.01536	CTG	102909	0.02218	0.01586
ATT	83398	0.01797	0.01489	CTT	63655	0.01372	0.01537

3-mers in Escherichia coli genome



Word	Count	Observed	Expected	Word	Count	Observed	Expected
GAA	83494	0.01800	0.01537	TAA	68838	0.01484	0.01490
GAC	54737	0.01180	0.01588	TAC	52592	0.01134	0.01539
GAG	42465	0.00915	0.01584	<i>TAG</i>	27243	0.00587	0.01536
GAT	86551	0.01865	0.01536	TAT	63288	0.01364	0.01489
GCA	96028	0.02070	0.01588	TCA	84048	0.01812	0.01539
GCC	92973	0.02004	0.01640	TCC	56028	0.01208	0.01589
GCG	114632	0.02471	0.01636	TCG	71739	0.01546	0.01586
GCT	80298	0.01731	0.01586	TCT	55472	0.01196	0.01537
GGA	56197	0.01211	0.01584	TGA	83491	0.01800	0.01536
GGC	92144	0.01986	0.01636	TGC	95232	0.02053	0.01586
GGG	47495	0.01024	0.01632	TGG	85141	0.01835	0.01582
GGT	74301	0.01601	0.01582	TGT	58375	0.01258	0.01534
GTA	52672	0.01135	0.01536	TTA	68828	0.01483	0.01489
GTC	54221	0.01169	0.01586	TTC	83848	0.01807	0.01537
GTG	66117	0.01425	0.01582	TTG	76975	0.01659	0.01534
GTT	82598	0.01780	0.01534	TTT	109831	0.02367	0.01487

2nd order Markov Chains



- Markov chains readily generalise to higher orders
- In 2nd order markov chain, position t depends on positions $t-1$ and $t-2$
- Transition matrix:

	A	C	G	T
AA				
AC				
AG				
AT				
CA				
...				

Codon Adaptation Index (CAI)



- Observation: cells prefer certain codons over others in highly expressed genes
 - Gene expression: DNA is transcribed into RNA (and possibly translated into protein)

Amino acid	Codon	Predicted	Gene class I	Gene class II	
Phe	TTT	0.493	0.551	0.291	Moderately expressed
	TTC	0.507	0.449	0.709	
Ala	GCT	0.246	0.145	0.275	Highly expressed
	GCC	0.254	0.276	0.164	
	GCA	0.246	0.196	0.240	
	GCG	0.254	0.382	0.323	
Asn	AAT	0.493	0.409	0.172	
	AAC	0.507	0.591	0.828	

Codon frequencies for some genes in *E. coli*

Codon Adaptation Index (CAI)



- Consider an amino acid sequence $X = x_1x_2\dots x_n$
- Let p_k be the probability that codon k is used in highly expressed genes
- Let q_k be the highest probability that a codon coding for the same amino acid as codon k has
 - For example, if codon k is "GCC", the corresponding amino acid is Alanine (see genetic code table; also GCT, GCA, GCG code for Alanine)
 - Assume that $p_{GCC} = 0.164$, $p_{GCT} = 0.275$, $p_{GCA} = 0.240$, $p_{GCG} = \mathbf{0.323}$
 - Now $q_{GCC} = q_{GCT} = q_{GCA} = q_{GCG} = \mathbf{0.323}$

Codon Adaptation Index (CAI)



- CAI is defined as

$$CAI = \left(\prod_{k=1}^n p_k / q_k \right)^{1/n}$$

- CAI can be given also in *log-odds* form:

$$\log(CAI) = (1/n) \sum_{k=1}^n \log(p_k / q_k)$$

CAI: example with an E. coli gene

 q_k
 p_k


M	A	L	T	K	A	E	M	S	E	Y	L	...	
ATG	GCG	CTT	ACA	AAA	GCT	GAA	ATG	TCA	GAA	TAT	CTG		
1.00	0.47	0.02	0.45	0.80	0.47	0.79	1.00	0.43	0.79	0.19	0.02		
	0.06	0.02	0.47	0.20	0.06	0.21		0.32	0.21	0.81	0.02		
	0.28	0.04	0.04		0.28			0.03			0.04		
	0.20	0.03	0.05		0.20			0.01			0.03		
		0.01						0.04			0.01		
		0.89						0.18			0.89		
ATG	GCT	TTA	ACT	AAA	GCT	GAA	ATG	TCT	GAA	TAT	TTA		
	GCC	TTG	ACC	AAG	GCC	GAG		TCC	GAG	TAC	TTG		
	GCA	CTT	ACA		GCA			TCA			CTT		
	GCG	CTC	ACG		GCG			TCG			CTC		
		CTA						AGT			CTA		
		CTG						AGC			CTG		
[1.00	0.20	0.04	0.04	0.80	0.47	0.79	1.00	0.03	0.79	0.19	0.89...]
[1.00	0.47	0.89	0.47	0.80	0.47	0.79	1.00	0.43	0.79	0.81	0.89]

CAI: properties



- CAI = 1.0 : each codon was the most frequently used codon in highly expressed genes
- Log-odds used to avoid numerical problems
 - What happens if you multiply many values <1.0 together?
- In a sample of E.coli genes, CAI ranged from 0.2 to 0.85
- CAI correlates with mRNA levels: can be used to predict high expression levels

Biological words: summary



- Simple 1-, 2- and 3-mer models can describe interesting properties of DNA sequences
 - GC skew can identify DNA replication origins
 - It can also reveal *genome rearrangement* events and *lateral transfer* of DNA
 - GC content can be used to locate genes: human genes are comparably GC-rich
 - CAI predicts high gene expression levels

Biological words: summary



- $k=3$ models can help to identify correct *reading frames*
 - Reading frame starts from a start codon and stops in a stop codon
 - Consider what happens to translation when a single extra base is introduced in a reading frame
- Also word models for $k > 3$ have their uses